ICT Express 11 (2025) 690–695

Contents lists available at ScienceDirect

ICT Express

journal homepage: www.elsevier.com/locate/icte

Lightweight federated learning-based intrusion detection system for industrial internet of things

Sun-Jin Lee, Il-Gu Lee *

Department of Future Convergence Technology Engineering, Sungshin Women's University, Seoul, 02844, Korea

ARTICLE INFO

Keywords:
Federated learning
Industrial internet of things
Intrusion detection

ABSTRACT

As machine learning technology advances, data security becomes increasingly important. In this study, we propose an intrusion detection mechanism based on federated learning (FL) that updates only the learning weights to minimize the risk of information leakage. Considering the limited resources of industrial Internet of Things (IIoT) nodes, we propose a learning method based on data pruning. The proposed FL-based intrusion detection model was found to be more secure than the centralized model in terms of the data leakage rate. Data pruning technology reduced the memory usage by 1.4 times while maintaining 97.7 % accuracy. The proposed method detects attacks in industrial sites where large-scale IIoT nodes are installed efficiently, and protects industrial secrets and personal information effectively.

1. Introduction

As a result of innovations in smart manufacturing technologies, the Internet of Things (IoT) is being installed in industrial sites for sensing purposes. Smart IoT nodes collect data through sensors and are used for core operation removal, optimization, risk detection, and monitoring in industries [1,2]. However, cyber hacking threats are also increasing because of vulnerabilities in industrial systems [3–5].

At present, industrial control systems use insecure authentication technologies and often fail to update security software regularly [3]. Attackers can exploit these weaknesses by modifying sensor logs and disabling risk alerts, increasing the likelihood of successful attacks [6]. Additionally, industrial IoT (IIoT) systems often operate in a centralized manner, enabling industrial equipment to be monitored and controlled from a central server. Thus, central servers are prime targets for attackers, who may attempt to leak or delete critical industrial information [7]. Therefore, future industrial control systems must strengthen security technologies. In addition, the centralized approach consumes a significant amount of resources [8]. An attack detection framework should be implemented to identify threats in advance [9].

We propose a federated learning (FL)-based attack detection framework that is designed for low power consumption and high efficiency. We use the MITRE ATT&CK strategy to classify attack techniques in an IIoT environment. The proposed framework enables each sensor to learn autonomously from collected data, transmitting only model weights to a central server. This approach minimizes the impact of hacking threats such as eavesdropping attacks, and prevents real-time data synchronization issues caused by central node overload. We also introduce a memory optimization method that employs a data pruning technique while considering the availability of each sensor node.

The remainder of this paper is organized as follows. Section 2 reviews research on security technologies for control systems. Section 3 presents the proposed FL-based attack detection framework. Section 4 describes the experimental setup and compares the proposed method with existing approaches. Finally, Section 5 concludes the paper.

2. Related work

The most common attack in industry is the advanced persistent threat (APT) attack. APT attacks involve leaking key confidential information and destroying systems within an industry through various types of attacks over a long period. MITRE ATT&CK tactics and techniques have been defined to establish and utilize various attack patterns for attack detection [10]. The MITRE ATT&CK framework organizes attack methods according to the cyberkill chain and divides them into threat strategies and techniques. The cyberkill chain comprises reconnaissance, weaponization and delivery, exploitation and installation, command and control, and action and exfiltration. MITRE ATT&CK categorizes these processes into 14 threat strategies and over 191 techniques [11]. Attackers can be identified in advance using strategy

* Corresponding author.
E-mail address: iglee@sungshin.ac.kr (I.-G. Lee).

https://doi.org/10.1016/j.icte.2025.05.002
Received 12 December 2024; Received in revised form 28 April 2025; Accepted 7 May 2025
Available online 9 May 2025

and technique classification, and the framework can detect and respond to attacks rapidly. The trends of conventional attack detection methodologies are shown in Table 1.

Conventional methods do not consider the resource-constrained environment of lightweight IoT nodes, and because all data are transmitted to the central node to learn and evaluate the data, this node might become overloaded. In addition, even if FL is used, the various types of attacks that can occur in industrial settings are not considered.

3. Proposed attack detection framework

Conventional intrusion detection systems (IDSs) for network attacks operate in a centralized model, collecting logs from each IoT node and transmitting the data to the server at a specific update interval. These data are subsequently used to build a model through centralized learning. However, this method increases the burden on the central server, making it difficult to deal with attackers who leak confidential industrial information during data transmission. Moreover, although these systems can identify the attack method, they often fail to determine the detailed process or pattern. Therefore, a system must be created such that even if an attacker leaks data, they should not be able to identify any confidential information. The effectiveness of the detection system must also be improved, and identifying additional attack types that may occur immediately following attack detection must be possible.

Table 1
Trend of conventional attack detection methodologies.

Ref.	Detection method	Contributions	Data leakage risk	Lightweight
[12]	Centralized passive detection	• Collected EDR logs and measured attack detection coverage in Carbanak environment	◦	×
[13]		• Collected EDR logs and measured attack detection coverage in APT29 environment	◦	×
[14]		• Demonstrated the feasibility of detecting RAASNet ransomware using EDR logs	◦	×
[15]		• Proved the feasibility of detecting APT attacks based on GRR and OSquery	◦	×
[16]	Centralized automatic detection	• Proposed an IDS optimization method using ensemble models	◦	×
[17]		• Proposed an IDS that simultaneously solves the hierarchical dependency of traffic attributes and the class imbalance problem	◦	◦
[18]		• Proposed a low-power IDS methodology based on feature selection and extraction	◦	×
[19]	Distributed automatic detection	• Proposed an FL-based IDS system for agricultural IoT	×	△
[20]		• Proposed a trust bonding mechanism to enhance the reliability of FL in digital twin mobile networks	◦	×
[21]		• Proposed a FedGDD-based malicious model detection technique that can defend against model poisoning attacks	◦	×

◦ = fully considered; △= partially considered; × = not considered.

This study proposes a lightweight FL-based intrusion detection framework, which serves as an attack classification system that enhances privacy by sharing only the weight values of the data, while preserving the performance of attack-type classification in an IIoT environment. In addition, MITRE ATT&CK has been implemented and is currently used to systematize tactics, techniques, and procedures. This system aids in understanding and defending against the workflow of an attacker, aligning the evolution of attack activities. The proposed framework and a flowchart thereof are shown in Fig. 1.

As shown in Fig. 1, the central server is pre-equipped with basic technique and tactic models that are trained in advance.

These technique classification models are then distributed to each node. At present, the basic technique serves as a weak classifier. Data from IIoT nodes are collected and stored in a distributed manner. Instead of transferring all log data that are collected from each node to the central server for learning, only the weight values are shared between the central server and IoT node. The model used in this study is federated averaging (FedAvg) [22], which is the most representative FL algorithm. In FedAvg, each terminal repeatedly learns a certain number of times and then transmits the parameter values to the server. FedAvg offers lower implementation complexity, fewer tuning parameters, and reduced client-side computation compared with advanced FL algorithms such as FedProx, making it particularly suitable for resource-constrained IIoT environments. Because the client learns by dividing the data by batch size, the convergence time of the global parameters can be shortened and a minibatch effect can be expected. This approach facilitates learning from the data collected by the IoT node. The updated model can immediately be used for detection and response at the endpoint level. Subsequently, the central server can update the global model using the parameters that are received from multiple IIoT nodes. This process enables the creation of a robust classification model. By limiting transmission to only the weight values, the leakage of data and personal information related to the core technology contained in the data can be prevented. The threat strategy and technique log data defined by the MITRE ATT&CK framework are used to collect log information.

The proposed detection framework employs data pruning to reduce memory usage in FL. Among the representative data pruning methods, including feature pruning [23], weight pruning is applied [24]. Weight pruning reduces the model size and minimizes overheads by removing connections between low-importance weight values, resulting in a sparser neural network. Specifically, individual weights in the weight matrix are set to zero, and the weights in $|W|$ are sorted by absolute value. The smallest sparsity k is then selected. This approach reduces the number of stored parameters and size of the model updates, leading to lower computational overhead and memory usage. Initially, pruning maintains accuracy as only unimportant weights are removed. However, if the pruning ratio exceeds a certain threshold, important weights may also be eliminated, leading to reduced accuracy. Therefore, determining and applying an optimal pruning ratio is essential for maintaining the model performance.

Algorithm 1 presents the pseudocode of the proposed framework. The framework operates by adding the pruning algorithm to the FedAvg algorithm. w denotes the weight value, k is the number of IIoT nodes, t is the time round, E is the number of local epochs, and B is the local minibatch size. At the server side, the basic model is created, and the weight value received from the IIoT node is then updated. At the IIoT node side, pruning work on the dataset is performed first, and then a local update is performed every time learning for one batch is completed. After this operation has progressed for E epochs, the final weights are sent to the central server for a global update [22].

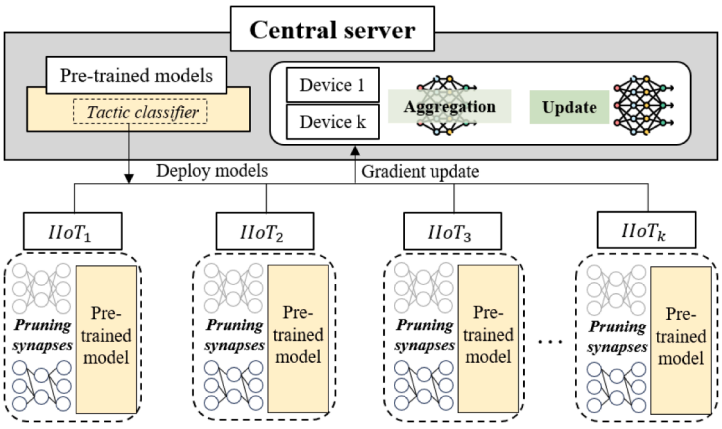

Fig. 1. Structure of the proposed framework.

Algorithm 1
Pseudocode of the proposed framework.

Server executes:
Generate basic_model
Initialize w_0
Send basic_model to IIoT node
for each round $t = 1, 2, \ldots$ **do**
for each IIoT node **in parallel do**
$w_{t+1}^k \leftarrow$ IIoT_Update(w_t)
IIoT node executes:
Collect sensing_data
Prune data to preset values
for each local epoch i from 1 to E **do**
for batch b $\in \mathscr{B}$ **do**
IIoT_LocalUpdate(w)
Return w to server

4. Experiments and analysis

4.1. Experimental environment

In this section, the performance of the proposed IIoT data security framework is evaluated. The endpoint detection and response (EDR) dataset that was used in the experiment was created by mapping the logs and MITRE ATT&CKT-IDs collected by performing an attack using the VAS Tool based on 114 selected threat strategies. The environment in which the dataset was collected is shown in Fig. 2. The attack server was built on Windows, Kali Linux, and SPiDER TM. Each process was run in a virtualized environment, and an intrusion prevention system (SNIPER ONE-I) and a web application firewall (WEBINSIGHT) were used. SPiDER TM AI was used to collect the logs. SPiDER TM is a security information and event-based management solution that integrates security control experience and big data utilization capabilities. SPiDER TM collects and stores all logs in real time and analyzes them with the latest threat information, such as harmful IPs and malicious URLs, thereby enabling efficient threat detection and prevention [25].

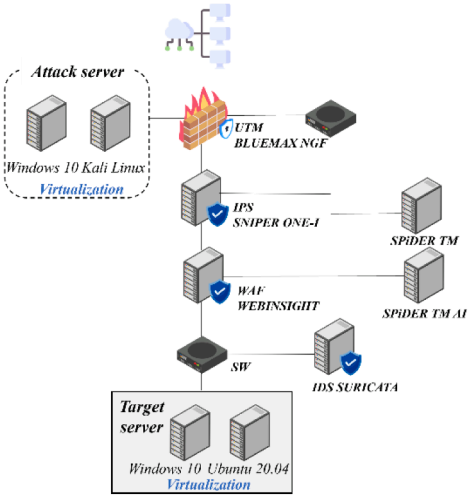

Fig. 2. Dataset collection environment.

The dataset contained 66 features. Each feature consisted of the process ID and time type, connection information, event information (order, classification, summary, and time), file attributes (name, path, and type), login information, protocol and registry information, detection rules, transmission and reception information, SHA256, and window information (class and title).

The classification label was divided into the MITRE ATT&CK Tactic ID and Technique ID. Only the Tactic ID was used in this experiment because the Technique ID consists of 79 items and is unsuitable for machine-learning classification. The Tactic ID consists of MITRE ATT&CK Tactics for numbers 1 to 10 and 40, as follows: Initial Access (TA0001), Execution (TA0002), Persistence (TA0003), Privilege Escalation (TA0004), Defense Evasion (TA0005), Credential Access (TA0006), Discovery (TA0007), Lateral Movement (TA0008), Collection (TA0009), Exfiltration (TA0010), and Impact (TA0040). The accuracy, memory utilization, latency, and security were simulated under the same environmental conditions for comparison with a conventional centralized model. The experimental setup is shown in Fig. 2. The evaluation was performed in an environment with an Intel(R) Core™ i7–1065G7 CPU @ 1.30 GHz, Intel(R) Iris(R) Plus Graphics, 16 GB of RAM, and Python version 3.10.12.

The proposed model calculates the weights through model learning for the datasets at each node, transmits them to the central server, and classifies them according to their average values. The experiment was conducted by slicing 64,278 out of 91,875 data points for the test set, excluding the data points mixed by row. We used the Sequential model of the Keras library in Python for each local learning process in the FL. The data in 48 rows were first transmitted to a hidden layer composed of 10 nodes and then to a hidden layer composed of five nodes. ReLU and softmax were used as the activation functions for the hidden and output layers, respectively. Optimization was performed using the Adam optimizer and the learning rate was set to 0.001. Considering the logarithm of the power value, the convergence speed increased as the error increased; categorical cross-entropy was used as the error function. Next, multiple workers shared the weights and the FedAvg model was used as the FL approach. The local model was trained for five epochs, following which the global model was updated by averaging the weights. Training was conducted with the number of nodes increasing from 5 to 50. In addition, the data accuracy was measured according to

the scarcity ratio and pruned. The set-weights function of the Keras library was used in the pruning technique.

The conventional model used in this study is a centralized model that collects and learns all data from one node to classify tactics [26]. Imran et al. [26] introduced a machine-learning algorithm to detect cybersecurity tactics, techniques, and procedures in industrial control systems (ICSs) rapidly. To define the current security posture of an organization, the ICS asset inventory, security baseline, network architecture, known vulnerabilities, and risk assessment were identified and threat cases were constructed on this basis. The authors mapped Windows event logs to MITRE ATT&CK tactics and then extracted features. There were a total of 10 threat scenarios. Each scenario included remote login, differential login permissions according to organizational policy, network service scanning, Windows audit log deletion, Windows firewall configuration change, Windows event log temper, and legitimate data. In addition, datasets were collected for case-by-case threat detection. Windows log events in the dataset included privilege escalation, discovery, execution, persistence, defense evasion, impair defense, and indicator removal on the host among APT tactics. Only some features were selected for ease of use. The selected characteristics included time, username, host, event code, signature, logon type, and process.

In this experiment, after preprocessing the dataset, the condition of collecting data sequentially by mixing the data by row was excluded, and the ratio of the training to test sets was set to 80:20. Additionally, during the model evaluation process, the model performance was optimized through hyperparameter tuning and model prediction was performed using the test dataset.

The method of [26] was used as the conventional centralized model, and a sequential model was implemented using the Keras library in Python to match the proposed model and experimental environment. The data from 52 rows were first transmitted to a hidden layer composed of 10 nodes and then to a hidden layer composed of five nodes. The activation and error functions used in the hidden and output layers were the same as those in the experimental environment for the proposed model.

4.2. Evaluation results and analysis

This section presents the accuracy, security, and memory usage of the proposed and conventional models.

4.2.1. Security

The security was measured using the data leakage rate according to the data leakage success rate of the attacker, as shown in Fig. 3(a).

It is assumed that one attack is performed in every round, and the number and intensity of attacks increase with the success rate of the attacker. An attack succeeds and results in data leakage if the probability of success is less than the attack success rate of the attacker. Conversely, if the probability is greater than the attack success rate, the attack fails and no data are leaked. In this case, the probability is proportional to the amount of intercepted data. It is assumed that the attacker eavesdrops on the communication between the node and central server. The data leakage rate of packets transmitted to the central server when an attacker attempts and succeeds in an attack can be expressed as a security index, as shown in Eq. (1).

$$\textit{Data leakage rate}\ (\%) = \frac{\textit{Leaked privacy data}}{\textit{Total privacy data}} \times 100 \tag{1}$$

To evaluate the security, 11 features containing IIoT patterns and confidential information were selected from the dataset. The conventional model has relatively weak security because, when data are leaked, all privacy information contained therein is also leaked. However, when using the proposed model, even if a data leakage attack is successful, only the weight values are shared, making it difficult to infer the information contained in the data, and predicting the personal information therein is even more difficult. Therefore, it was assumed that only partial information could be predicted using a probabilistic model. The dataset used in the experiment was approximately 30 MB in total. Hence, the conventional model could leak 5.8 MB of confidential information in 90 % of the environments where data leakage attempts were most frequent. However, the proposed model only leaked a maximum of 0.6 MB even in 90 % of the environments, which was a much smaller loss. This could reduce the rate of obtaining confidential information and minimize re-identification.

4.2.2. Latency

Fig. 3(b) depicts the delay of each model. The conventional model measured the time required for learning and evaluation, whereas the proposed model measured the local learning time and update and evaluation times of each node.

When the number of nodes was small, the time difference between the proposed and conventional models was the largest. As the number of nodes increased, the number of nodes used for classification increased. Therefore, attacks were detected with low latency. The delay in the conventional model was greater than that in the proposed model. When the number of nodes was 5, the conventional model had a delay of approximately 367 s, whereas the proposed model had a delay of approximately 15.8 s, which was approximately 23.3 times faster. Even in an environment with 50 nodes, the proposed model exhibited an improvement of approximately 25.2 times.

Thus, when using the proposed model, attacks can be detected faster and with less computational complexity than when using conventional models by learning data collected from individual nodes on their own and quickly collecting evaluation results from the central server. In addition, the proposed method can learn and evaluate models with low latency without being limited by speed, even in an environment in which the number of nodes increases.

4.2.3. Accuracy

The accuracy was measured using Eq. (2) to address potential overfitting when compiling the model by excluding the test set from the learning process. In this context, accuracy represents the model performance evaluated using the test dataset after the weight update.

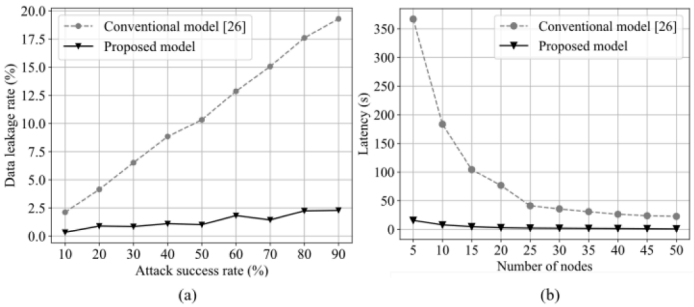

Fig. 3. (a) Data leakage rate based on the attack success rate and (b) latency comparison between the proposed and conventional models.

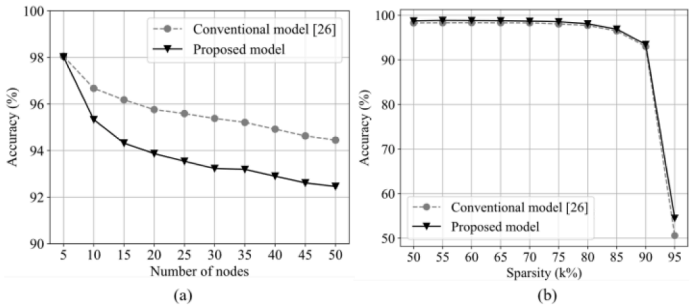

Fig. 4. Performance evaluation results based on (a) the number of nodes and (b) sparsity.

$$Accuracy\ (\%) = \frac{Correctly\ classified\ (predicted)\ data}{Total\ data} \times 100 \quad (2)$$

Fig. 4(a) shows the accuracies of the proposed and conventional models as a function of the number of nodes. As the number of nodes increased, the difference in accuracy between the proposed and conventional models gradually increased and then converged to approximately 2 %. When five nodes were used for the tactical classification, the proposed and conventional models achieved accuracies of 98 % and 98.1 %, respectively. When the number of nodes increased to 50, the accuracy of the proposed model decreased to 92.5 %, whereas the conventional model maintained an accuracy of 94.5 %. Despite this, the proposed model showed only a 2 % difference in accuracy compared with the conventional model. This demonstrates that our model can classify attacks with comparable accuracy while offering enhanced security relative to the existing centralized model.

The Keras library improves the classification performance through pruning, even with a small amount of data. The weight pruning can be optimized by setting the target sparsity (k-sparsity), where k is a percentage of the weight and is set to zero. The weights were ranked and the smallest value was set to zero according to the size of the weight $|W|$. The accuracies of the conventional and proposed models using the data pruning technique are shown in Fig. 4(b).

The data size decreased as the data sparsity increased, regardless of the number of nodes, resulting in low accuracy and memory utilization. Following pruning by increasing the data rate by 10 %, we measured the accuracy and memory usage at the four ratios with the highest k-sparsity (75 %, 80 %, 85 %, and 90 %) for a more detailed analysis.

According to Fig. 5, the accuracy was the highest when the k-sparsity was set to 80 %. The memory usage rate was the lowest at 75 % and similar to that at 80 %. Therefore, we set the optimal k-sparsity ratio of the proposed model to 80 %.

4.2.4. Memory usage

Fig. 6 shows the memory usage considering the model size, and computational and communication overheads. The conventional model without pruning exhibited a memory usage of 50.1 MB in an environment with five nodes and 287.6 MB in an environment with 50 nodes, whereas the proposed model without pruning showed a memory usage of 109.1 and 624.2 MB, respectively.

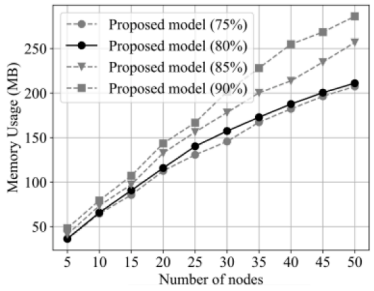

Fig. 6. Memory usage comparison between the conventional and proposed models.

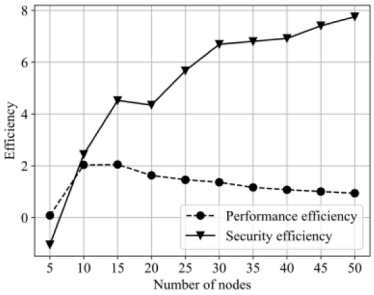

Fig. 7. Optimal point between performance and security.

That is, the memory usage rate of the proposed model without pruning was 2.2 times higher than that of the conventional model. The increased memory usage is because, in an FL-based model, each node has a parallel format that processes data quickly using a large amount of memory simultaneously. Conversely, the proposed model with 80 % pruning showed improved memory usage compared with the conventional model, at 36.4 MB in an environment with five nodes and 211.1 MB in an environment with 50 nodes. In addition, the accuracy was maintained at 97.7 %; therefore, the memory utilization could be maximized while minimizing the decrease in accuracy. Thus, the

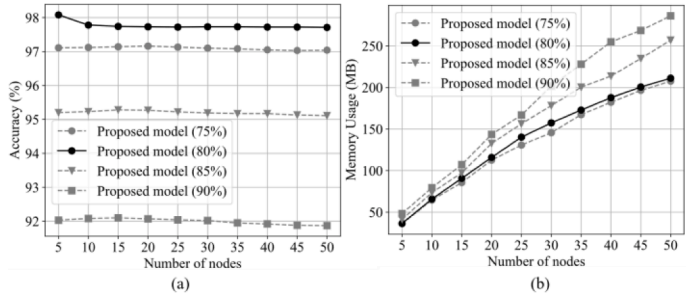

Fig. 5. Performance evaluation results in terms of the (a) accuracy and (b) memory usage.

proposed model with pruning applied has higher usability in IIoT environments with limited resources compared with other models.

Fig. 7 shows the optimal point between the performance and security of the proposed model. To measure the efficiency, the performance and security of the conventional and proposed models were divided by complexity, and the difference was then calculated and visualized. A general deterioration in the performance efficiency was observed as the number of nodes increased. However, the security efficiency showed a pattern of increasing as the number of nodes increased. This is because the security of the conventional model decreased, whereas that of the proposed model decreased slightly. The experiment confirmed that the performance of the proposed model was optimized at approximately 10 nodes, where the two graphs intersected.

5. Conclusions

As the number of IoT nodes used in industry and the value of the information collected, stored, and transmitted by the IIoT increase, the likelihood of the IIoT becoming a major target for attackers increases. Therefore, lightweight FL-based intrusion detection is a desirable feature in future IIoT security frameworks. By introducing the proposed model, security can be guaranteed while maintaining the classification model accuracy, thereby preparing for various threats that occur at industrial sites. In particular, industrial organizations that handle key confidential information must be able to utilize the collected log information safely and detect attacks rapidly. Using the proposed FL-based low-power detection system, the accuracy can be maintained at 97.7 %. Even when the attack success rate increases, the model achieves approximately 5.2 MB less data leakage compared with the conventional model, and the memory usage can be reduced by a factor of approximately 1.4, making it a realistic model for industrial applications. However, the proposed model only measures the classification performance for the MITRE ATT&CK type and cannot distinguish between normal and attack logs. In future research, we plan to develop a real-world test environment with CPU and memory configurations, expand the attack scenarios and security strategies, and refine the evaluation metrics. In addition, we will consider explainable AI techniques and examine scalability, latency, and hardware constraints to enhance the practicality of the proposed model in IIoT systems.

CRediT authorship contribution statement

Sun-Jin Lee: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Data curation, Conceptualization. **Il-Gu Lee:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Ministry of Trade, Industry and Energy (MOTIE) under Training Industrial Security Specialist for High-Tech Industry (RS-2024–00415520) supervised by the Korea Institute for Advancement of Technology (KIAT), and the Ministry of Science and ICT (MSIT) under the ICAN (ICT Challenge and Advanced Network of HRD) program (IITP-2022-RS-2022–00156310) and Information Security Core Technology Development (RS-2024–00437252) supervised by the Institute of Information & Communication Technology Planning & Evaluation (IITP).

References

[1] R. Ahmad, et al., A comprehensive deep learning benchmark for IoT IDS, Comput. Secur. 114 (2022) 102588, https://doi.org/10.1016/j.cose.2021.102588.
[2] M. Soori, B. Arezoo, R. Dastres, Internet of things for smart factories in industry 4.0: a review, Internet Things Cyber-Phys. Syst. 3 (2023) 192–204, https://doi.org/10.1016/j.iotcps.2023.04.006.
[3] A.M.Y. Koay, et al., Machine learning in industrial control system (ICS) security: current landscape, opportunities and challenges, J. Intell. Inf. Syst. 60 (2023) 377–405, https://doi.org/10.1007/s10844-022-00753-1.
[4] R.J. Raimundo, A.T. Rosário, Cybersecurity in the internet of things in industrial management, Appl. Sci. 12 (2022) 1598, https://doi.org/10.3390/app12031598.
[5] T. Saba, et al., Anomaly-based intrusion detection system for IoT networks through deep learning model, Comput. Electr. Eng. 99 (2022) 107810.
[6] M. Mesbah, et al., Analysis of ICS and SCADA systems attacks using honeypots, Fut. Int 15 (2023) 241, https://doi.org/10.3390/fi15070241.
[7] C. Kaura, N. Sindhwani, A. Chaudhary, Analysing the impact of cyber-threat to ICS and SCADA systems, in: Proceedings of the 2022 International Mobile and Embedded Technology Conference (MECON), Noida, India, 2022, pp. 466–470, https://doi.org/10.1109/MECON53876.2022.9752425.
[8] Z. Jin, et al., FL-IIDS: a novel federated learning-based incremental intrusion detection system, Fut. Gener. Comput. Syst. 151 (2024) 57–70.
[9] E.M. Onyema, et al., Design of intrusion detection system based on cyborg intelligence for security of cloud network traffic of smart cities, J. Cloud Comput. 11 (2022) 14–20.
[10] B. Al-Sada, A. Sadighian, G. Oligeri, MITRE ATT&CK: state of the art and way forward, ACM Comput. Surv. 57 (2025) 1–37, https://doi.org/10.1145/3687300.
[11] A. Virkud, M.A. Inam, A. Riddle, J. Liu, G. Wang, A. Bates, How does endpoint detection use the MITRE ATT&CK framework?, in: Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24), 2024, pp. 3891–3908, https://doi.org/10.5555/3698900.3699118.
[12] S.E. Jeon, et al., An effective threat detection framework for advanced persistent cyberattacks, CMC Comput. Mater. Contin. 75 (2023) 4231–4253, https://doi.org/10.32604/cmc.2023.034287.
[13] N.E. Park, et al., Performance evaluation of a fast and efficient intrusion detection framework for advanced persistent threat-based cyberattacks, Comput. Electr. Eng. 105 (2023) 108548, https://doi.org/10.1016/j.compeleceng.2022.108548.
[14] S.-J. Lee, et al., Ransomware detection using open-source tools, in: Proceedings of the 24th International Conference on Advanced Communication Technology (ICACT), Pyeongchang, Republic of Korea, 2022, pp. 1385–1391, https://doi.org/10.23919/ICACT53585.2022.9728873.
[15] S.-H. Park, et al., Performance evaluation of open-source endpoint detection and response combining Google Rapid Response and Osquery for threat detection, IEEE Access 10 (2022) 20259–20269, https://doi.org/10.1109/ACCESS.2022.3152574.
[16] A. Alhowaide, I. Alsmadi, J. Tang, Ensemble detection model for IoT IDS, Int. Things 16 (2021) 100435, https://doi.org/10.1016/j.iot.2021.100435.
[17] W. Zhou, C. Xia, T. Wang, X. Liang, W. Lin, X. Li, S. Zhang, HIDIM: a novel framework of network intrusion detection for hierarchical dependency and class imbalance, Comput. Secur 148 (2025) 104155.
[18] K. Albulayhi, et al., IoT intrusion detection using machine learning with a novel high performing feature selection method, Appl. Sci. 12 (2022) 5015, https://doi.org/10.3390/app12105015.
[19] O. Friha, et al., FELIDS: federated learning-based intrusion detection system for agricultural Internet of Things, J. Parallel Distrib. Comput. 165 (2022) 17–31, https://doi.org/10.1016/j.jpdc.2022.03.003.
[20] J. Guo, et al., TFL-DT: a trust evaluation scheme for federated learning in digital twin for mobile networks, IEEE J. Sel. Areas Commun. 41 (2023) 3548–3560.
[21] C. Li, et al., RFL-APIA: a comprehensive framework for mitigating poisoning attacks and promoting model aggregation in IIoT federated learning, IEEE Trans. Ind. Inf. 20 (2024) 12935–12944, https://doi.org/10.1109/TII.2024.3431020.
[22] B. McMahan, et al., Communication-efficient learning of deep networks from decentralized data, PMLR (2017). Available: https://arxiv.org/abs/1602.05629.
[23] J. Carter, S. Mancoridis, E. Galinkin, Fast, lightweight IoT anomaly detection using feature pruning and PCA, in: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, 2022, pp. 133–138, https://doi.org/10.1145/3477314.3508377.
[24] W. Niu, et al., PatDNN: achieving real-time DNN execution on mobile devices with pattern-based weight pruning, in: Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems, 2020, pp. 907–922, https://doi.org/10.1145/3373376.3378534.
[25] A. Nascita, et al., Machine and deep learning approaches for IoT attack classification, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE Publications Infocom (IEEE Publications), 2022, https://doi.org/10.1109/INFOCOMWKSHPS54753.2022.9797971.
[26] M. Imran, et al., A performance overview of machine learning-based defense strategies for advanced persistent threats in industrial control systems, Comput. Secur. 134 (2023) 103445.